Target Case study

Red ones are the query

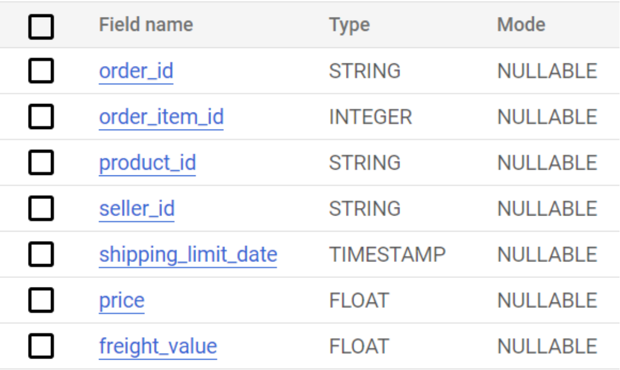
Then screenshot

Green ones are insights or suggestions.

Q1. Import the dataset and do usual exploratory analysis steps like checking the structure & characteristics of the dataset

a. Data type of columns in a table

ans:



□	Field name	Type	Mode
□	order_id	STRING	NULLABLE
□	order_item_id	INTEGER	NULLABLE
□	product_id	STRING	NULLABLE
□	seller_id	STRING	NULLABLE
□	shipping_limit_date	TIMESTAMP	NULLABLE
□	price	FLOAT	NULLABLE
□	freight_value	FLOAT	NULLABLE

Data type present in columns are string, integer, float, timestamp.

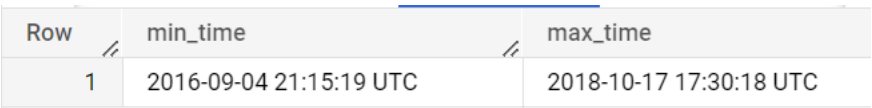
****

b. Time period for which the data is given

ans:

select min(order_purchase_timestamp) as min_time, max(order_purchase_timestamp) as max_time,

from `Target_store.orders`



Row	min_time	max_time
1	2016-09-04 21:15:19 UTC	2018-10-17 17:30:18 UTC

Time period for which the data is given was between 2016 to 2018

****

c. Cities and States of customers ordered during the given period

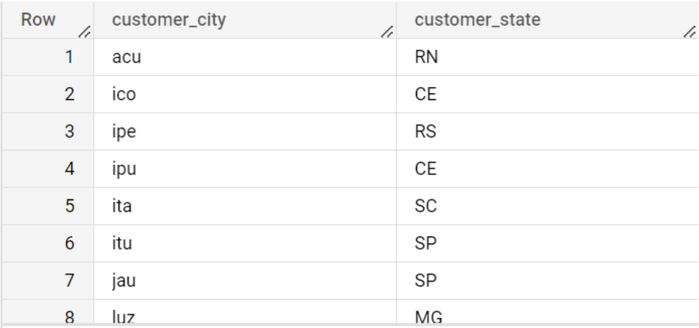
ans:

select distinct c.customer_city, c.customer_state

from `Target_store.customers` c join `Target_store.orders` o

on c.customer_id = o.customer_id

limit 10

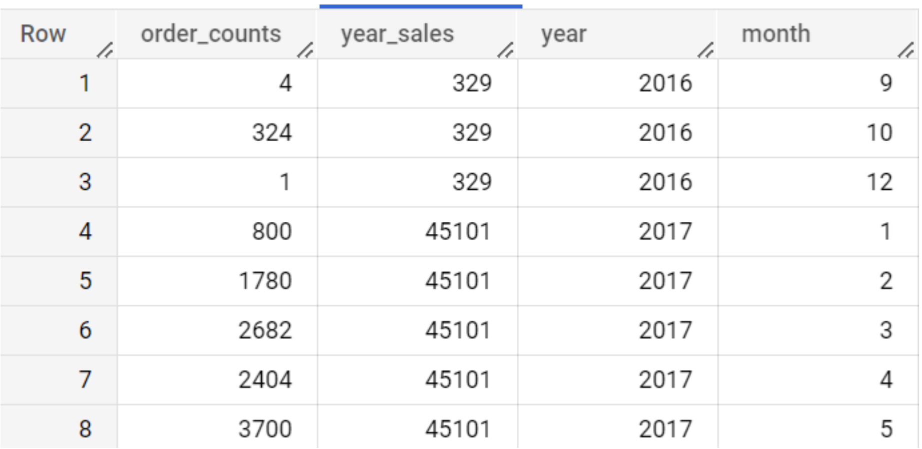


Row	customer_city	customer_state
1	acu	RN
2	ico	CE
3	ipe	RS
4	ipu	CE
5	ita	SC
6	itu	SP
7	jau	SP
8	luz	MG

Q2. In-depth Exploration:

a. Is there a growing trend on e-commerce in Brazil? How can we describe a complete scenario? Can we see some seasonality with peaks at specific months?
Ans:

```
with Q1 as
(select order_id, extract(month from order_purchase_timestamp) as month , extract(year from
order_purchase_timestamp) as year
from `Target_store.orders`)
select count (distinct order_id) as order_counts, sum(count (distinct order_id)) over(partition by
year) as year_sales , year, Q1.month from Q1
group by year, Q1.month
order by year, month
```



Row	order_counts	year_sales	year	month
1	4	329	2016	9
2	324	329	2016	10
3	1	329	2016	12
4	800	45101	2017	1
5	1780	45101	2017	2
6	2682	45101	2017	3
7	2404	45101	2017	4
8	3700	45101	2017	5

There was a growing trend on year on year basis as the no. of sales increased, In Nov 2017, Jan 2018, and March 2018 the sales were at the peak.

****

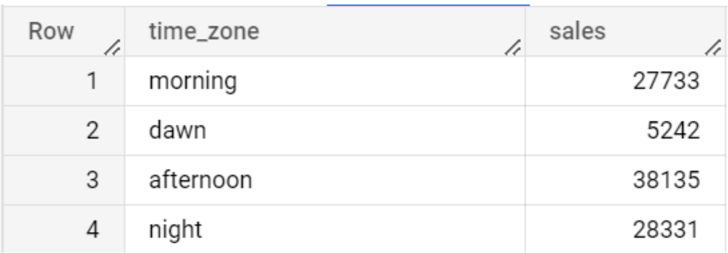
b. What time do Brazilian customers tend to buy (Dawn, Morning, Afternoon or Night)?
Dawn 12 am - 6 am
Morning 6 am - 12 pm
Afternoon 12 pm - 6 pm
Night 6-12 am
Ans:

```
with Q1 as
(select order_id, extract(hour from order_purchase_timestamp) as hours
from `Target_store.orders`),

Q2 as
(select case  when hours between 0 and  6 then "dawn"
          when hours between 6 and 12 then "morning"
          when hours between 12 and 18 then "afternoon"
          else "night"
```

```
         end as time_zone,  order_id
from Q1)

select time_zone, count(order_id) as sales from Q2
group by time_zone
```



Row	time_zone	sales
1	morning	27733
2	dawn	5242
3	afternoon	38135
4	night	28331

As we can see above data, Brazilian customers tend to buy at afternoon time

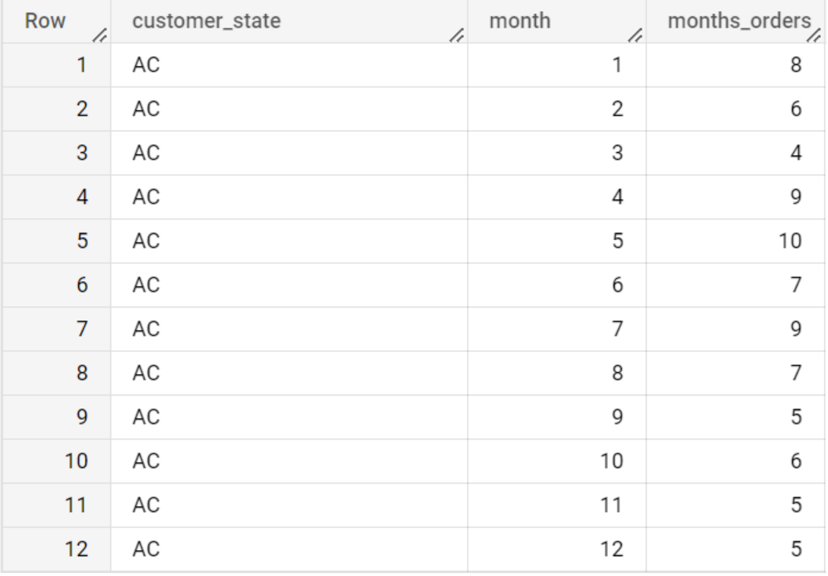
Q3. Evolution of E-commerce orders in the Brazil region:

a. Get month on month orders by states

ans:

```
with Q1 as
(select c.customer_state, o.order_id, extract(month from o.order_purchase_timestamp) as month
from `Target_store.customers` c join `Target_store.orders` o on c.customer_id = o.customer_id)

select customer_state,  month, count(order_id) as months_orders from Q1
group by  customer_state, month
order by  customer_state, month
```



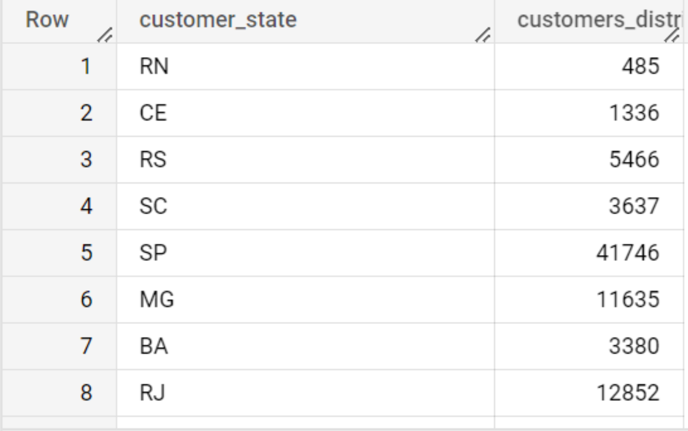
Row	customer_state	month	months_orders
1	AC	1	8
2	AC	2	6
3	AC	3	4
4	AC	4	9
5	AC	5	10
6	AC	6	7
7	AC	7	9
8	AC	8	7
9	AC	9	5
10	AC	10	6
11	AC	11	5
12	AC	12	5

****

b. Distribution of customers across the states in Brazil

ans:

```
select customer_state, count(customer_unique_id) as customers_distribution
from `Target_store.customers`
group by customer_state
```



Row	customer_state	customers_distr
1	RN	485
2	CE	1336
3	RS	5466
4	SC	3637
5	SP	41746
6	MG	11635
7	BA	3380
8	RJ	12852

These are the distribution for the customers present in each state.

Q4. Impact on Economy: Analyze the money movement by e-commerce by looking at order prices, freight and others.

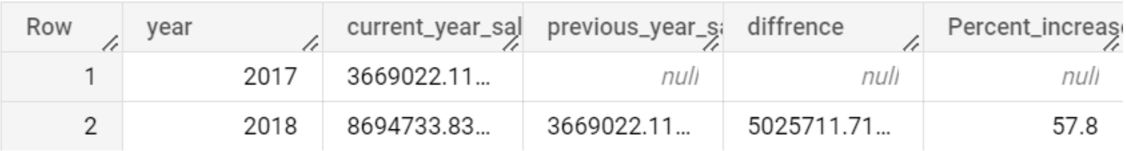
a. Get % increase in cost of orders from 2017 to 2018 (include months between Jan to Aug only) - You can use "payment_value" column in payments table

Ans:

```
with Q1 as
(
select extract(month from o.order_purchase_timestamp) as month,
extract(year from o.order_purchase_timestamp) as year,
o.order_id,
p.payment_value
from `Target_store.orders` o join `Target_store.payments` p on o.order_id = p.order_id
where extract(year from o.order_purchase_timestamp) = 2017 or extract(year from
o.order_purchase_timestamp)=2018
),

Q2 as
(select year, sum(payment_value) as current_year_sales, lag(sum (payment_value)) over(order by
year) as previous_year_sales,
sum(payment_value) - lag(sum (payment_value)) over(order by year) as diffrence,
 from Q1
where  month between 1 and 8
group by year
order by year)

select year, current_year_sales, previous_year_sales, diffrence,
round(diffrence/current_year_sales*100,2) as Percent_increase
from Q2
```



Row	year	current_year_sal	previous_year_s	diffrence	Percent_increas
1	2017	3669022.11...	null	null	null
2	2018	8694733.83...	3669022.11...	5025711.71...	57.8

There is 57.8% increase in cost of orders from 2017 to 2018.

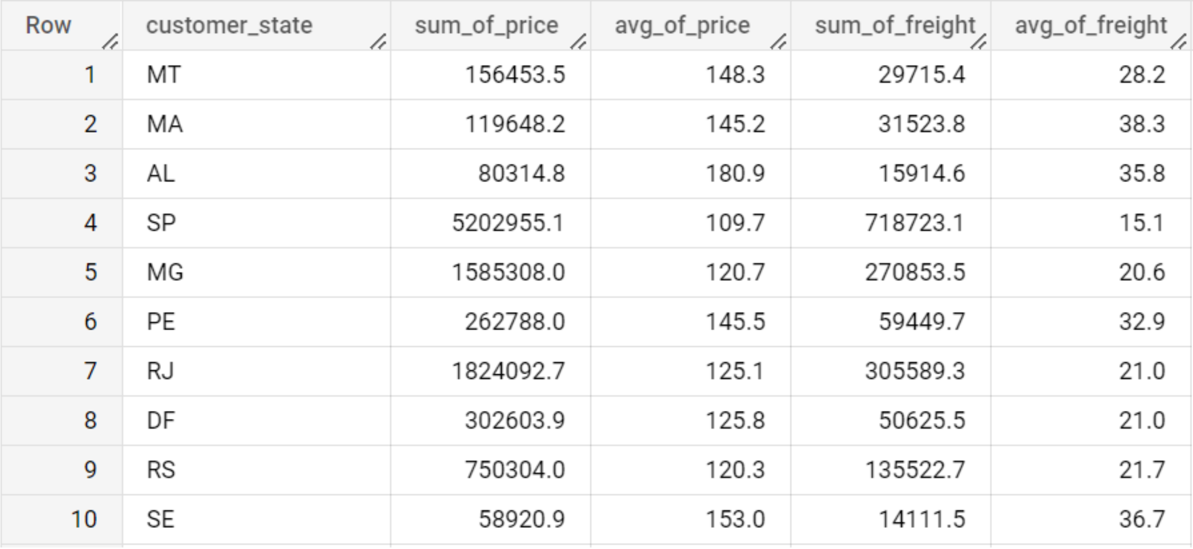
****

b. Mean & Sum of price and freight value by customer state

Ans:

```
select c.customer_state, round(sum(ot.price),1) sum_of_price, round(avg(ot.price),1) avg_of_price,
round(sum(ot.freight_value),1) sum_of_freight, round(avg(ot.freight_value),1) avg_of_freight from
`Target_store.customers` c
join `Target_store.orders` o on c.customer_id = o.customer_id
join `Target_store.order_items` ot on o.order_id = ot.order_id
```

group by c.customer_state



Row	customer_state	sum_of_price	avg_of_price	sum_of_freight	avg_of_freight
1	MT	156453.5	148.3	29715.4	28.2
2	MA	119648.2	145.2	31523.8	38.3
3	AL	80314.8	180.9	15914.6	35.8
4	SP	5202955.1	109.7	718723.1	15.1
5	MG	1585308.0	120.7	270853.5	20.6
6	PE	262788.0	145.5	59449.7	32.9
7	RJ	1824092.7	125.1	305589.3	21.0
8	DF	302603.9	125.8	50625.5	21.0
9	RS	750304.0	120.3	135522.7	21.7
10	SE	58920.9	153.0	14111.5	36.7

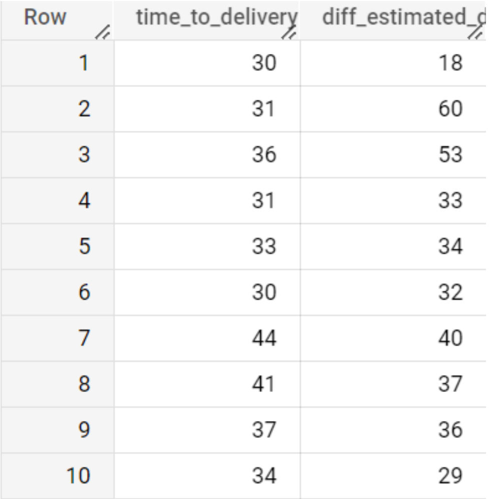
Q5. Analysis on sales, freight and delivery time

a. Calculate days between purchasing, delivering and estimated delivery

Ans:

```
with Q1 as
(select extract(date from order_purchase_timestamp) purchase_date, extract(date from
order_delivered_customer_date) delivery_date, extract(date from order_estimated_delivery_date)
estimate_deliv_date
from `Target_store.orders`)

select date_diff(delivery_date, purchase_date, day) as time_to_delivery,
date_diff(estimate_deliv_date, purchase_date, day) as diff_estimated_delivery
from Q1
where delivery_date is not null
```



Row	time_to_delivery	diff_estimated_d
1	30	18
2	31	60
3	36	53
4	31	33
5	33	34
6	30	32
7	44	40
8	41	37
9	37	36
10	34	29

There were many nulls in delivery date, which may be that orders was cancelled by the customer or some problems with the seller, but seems like some of the orders have over passed the estimated date, but many of the orders are delivered within the estimated date also.

****

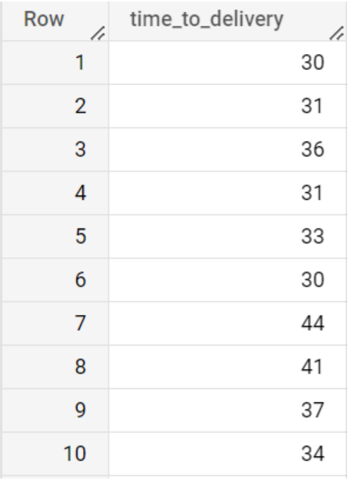
b. Find time_to_delivery & diff_estimated_delivery. Formula for the same given below:

b1. time_to_delivery = order_purchase_timestamp-order_delivered_customer_date

Ans:

```
with Q1 as
(select extract(date from order_purchase_timestamp) purchase_date, extract(date from
order_delivered_customer_date) delivery_date, extract(date from
order_estimated_delivery_date) estimate_deliv_date
from `Target_store.orders`)

select date_diff(delivery_date, purchase_date, day) as time_to_delivery
from Q1
where delivery_date is not null
```



Row	time_to_delivery
1	30
2	31
3	36
4	31
5	33
6	30
7	44
8	41
9	37
10	34

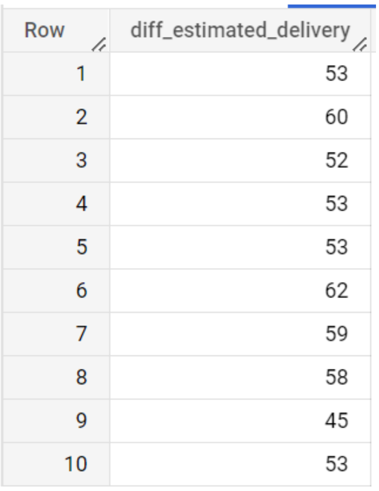
****

b2. diff_estimated_delivery = order_estimated_delivery_date-order_delivered_customer_date

Ans:

```
with Q1 as
(select extract(date from order_purchase_timestamp) purchase_date, extract(date from
order_delivered_customer_date) delivery_date, extract(date from
order_estimated_delivery_date) estimate_deliv_date
from `Target_store.orders`)

select
date_diff(estimate_deliv_date, purchase_date, day) as diff_estimated_delivery
from Q1
where delivery_date is not null
```



Row	diff_estimated_delivery
1	53
2	60
3	52
4	53
5	53
6	62
7	59
8	58
9	45
10	53

****

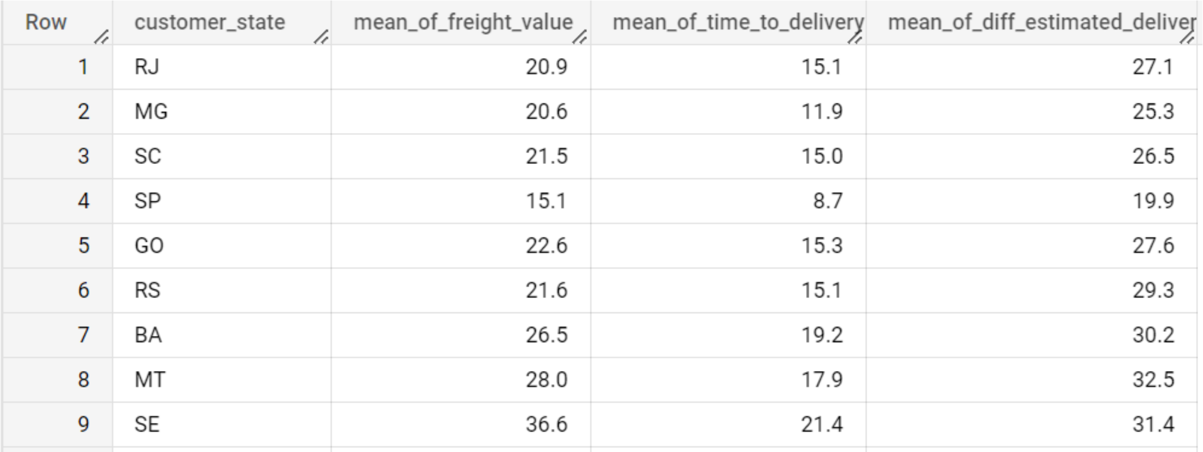
c. Group data by state, take mean of freight_value, time_to_delivery, diff_estimated_delivery

Ans:

```
with Q1 as
(select extract(date from order_purchase_timestamp) purchase_date, extract(date from
order_delivered_customer_date) delivery_date, extract(date from order_estimated_delivery_date)
estimate_deliv_date,   c.customer_state,
 ot.freight_value
from `Target_store.orders` o join `Target_store.order_items` ot on o.order_id = ot.order_id
join `Target_store.customers` c on o.customer_id = c.customer_id),

Q2 as
(select customer_state, date_diff(delivery_date, purchase_date, day) as time_to_delivery,
date_diff(estimate_deliv_date, purchase_date, day) as diff_estimated_delivery,
freight_value from Q1
where delivery_date is not null)

select customer_state, round(avg(freight_value),1) as mean_of_freight_value,
 round(avg(time_to_delivery),1)  as mean_of_time_to_delivery ,
 round(avg(diff_estimated_delivery),1) as mean_of_diff_estimated_delivery from Q2
group by customer_state
```



Row	customer_state	mean_of_freight_value	mean_of_time_to_delivery	mean_of_diff_estimated_deliver
1	RJ	20.9	15.1	27.1
2	MG	20.6	11.9	25.3
3	SC	21.5	15.0	26.5
4	SP	15.1	8.7	19.9
5	GO	22.6	15.3	27.6
6	RS	21.6	15.1	29.3
7	BA	26.5	19.2	30.2
8	MT	28.0	17.9	32.5
9	SE	36.6	21.4	31.4

This is the comparison of average freight value, average delivery time in days and average estimate delivery time.

****

d. Sort the data to get the following:

e. Top 5 states with highest/lowest average freight value - sort in desc/asc limit 5

Ans

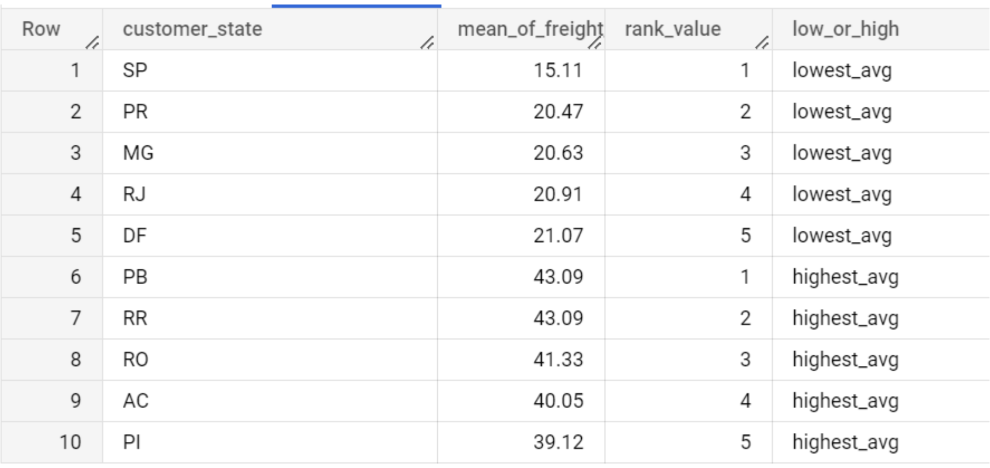
```
with Q1 as
(select extract(date from order_purchase_timestamp) purchase_date, extract(date from
order_delivered_customer_date) delivery_date, extract(date from order_estimated_delivery_date)
estimate_deliv_date,   c.customer_state,
 ot.freight_value
from `Target_store.orders` o join `Target_store.order_items` ot on o.order_id = ot.order_id
join `Target_store.customers` c on o.customer_id = c.customer_id),

Q2 as
(select customer_state, date_diff(delivery_date, purchase_date, day) as time_to_delivery,
date_diff(estimate_deliv_date, purchase_date, day) as diff_estimated_delivery,
freight_value from Q1
where delivery_date is not null),

Q3 as
(select customer_state, round(avg(freight_value),2) as mean_of_freight_value, dense_rank()
over(order by (avg(freight_value)) desc) as rank_value, "highest_avg" as low_or_high
from Q2
group by customer_state
order by avg(freight_value) desc
limit 5),

Q4 as
(select customer_state, round(avg(freight_value),2) as mean_of_freight_value, dense_rank()
over(order by (avg(freight_value)) ) as rank_value, "lowest_avg" as low_or_high
from Q2
group by customer_state
order by avg(freight_value)
limit 5)

select * from Q3
union all
select * from Q4
```



Row	customer_state	mean_of_freight	rank_value	low_or_high
1	SP	15.11	1	lowest_avg
2	PR	20.47	2	lowest_avg
3	MG	20.63	3	lowest_avg
4	RJ	20.91	4	lowest_avg
5	DF	21.07	5	lowest_avg
6	PB	43.09	1	highest_avg
7	RR	43.09	2	highest_avg
8	RO	41.33	3	highest_avg
9	AC	40.05	4	highest_avg
10	PI	39.12	5	highest_avg

This is the data of top 5 highest and lowest states which take average freight value for the product.

****

f. Top 5 states with highest/lowest average time to delivery

Ans:

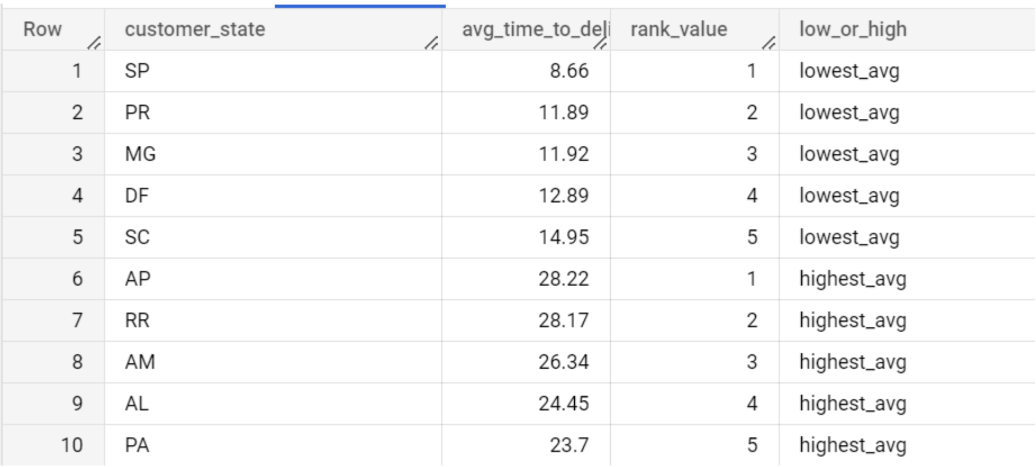
```
with Q1 as
(select extract(date from order_purchase_timestamp) purchase_date, extract(date from
order_delivered_customer_date) delivery_date, extract(date from order_estimated_delivery_date)
estimate_deliv_date,   c.customer_state,
 ot.freight_value
from `Target_store.orders` o join `Target_store.order_items` ot on o.order_id = ot.order_id
join `Target_store.customers` c on o.customer_id = c.customer_id),

Q2 as
(select customer_state, date_diff(delivery_date, purchase_date, day) as time_to_delivery,
date_diff(estimate_deliv_date, purchase_date, day) as diff_estimated_delivery,
freight_value from Q1
where delivery_date is not null),

Q3 as
(select customer_state, round(avg(time_to_delivery),2) as avg_time_to_delivery, dense_rank()
over(order by (avg(time_to_delivery)) desc) as rank_value, "highest_avg" as low_or_high
from Q2
group by customer_state
order by avg(time_to_delivery) desc
limit 5),

Q4 as
(select customer_state, round(avg(time_to_delivery),2) as avg_time_to_delivery, dense_rank()
over(order by (avg(time_to_delivery)) ) as rank_value, "lowest_avg" as low_or_high
from Q2
group by customer_state
order by avg(time_to_delivery)
limit 5)

select * from Q3
union all
select * from Q4
```



Row	customer_state	avg_time_to_deli	rank_value	low_or_high
1	SP	8.66	1	lowest_avg
2	PR	11.89	2	lowest_avg
3	MG	11.92	3	lowest_avg
4	DF	12.89	4	lowest_avg
5	SC	14.95	5	lowest_avg
6	AP	28.22	1	highest_avg
7	RR	28.17	2	highest_avg
8	AM	26.34	3	highest_avg
9	AL	24.45	4	highest_avg
10	PA	23.7	5	highest_avg

This is the data of top 5 highest and lowest states which take average time to deliver the product in days. There is more scope of improvement time taken to deliver in states with highest avg time.

****

g.Top 5 states where delivery is really fast/ not so fast compared to estimated date

Ans:

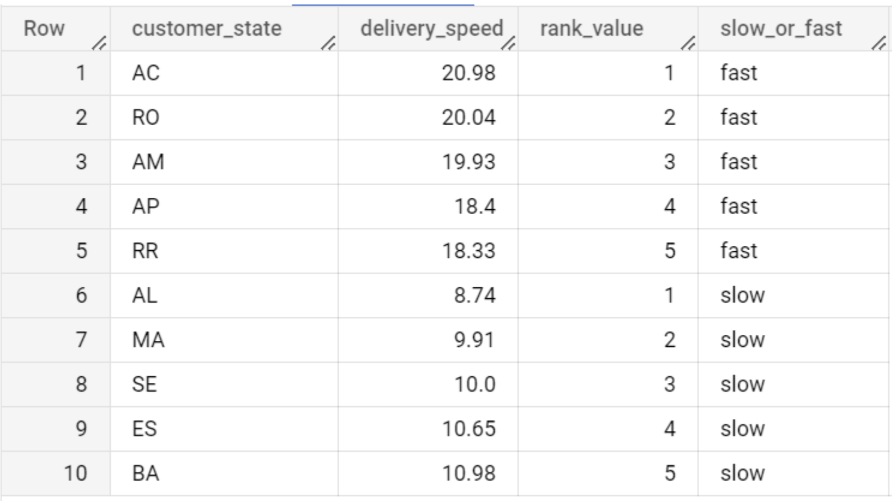
```
with Q1 as
(select extract(date from order_purchase_timestamp) purchase_date, extract(date from
order_delivered_customer_date) delivery_date, extract(date from order_estimated_delivery_date)
estimate_deliv_date,   c.customer_state,
 ot.freight_value
from `Target_store.orders` o join `Target_store.order_items` ot on o.order_id = ot.order_id
join `Target_store.customers` c on o.customer_id = c.customer_id),

Q2 as
(select customer_state, date_diff(delivery_date, purchase_date, day) as time_to_delivery,
date_diff(estimate_deliv_date, purchase_date, day) as diff_estimated_delivery,
freight_value from Q1
where delivery_date is not null),

Q3 as
(select customer_state, round(avg(diff_estimated_delivery - time_to_delivery),2) as delivery_speed,
dense_rank() over(order by (avg(diff_estimated_delivery - time_to_delivery)) desc) as rank_value,
"fast" slow_or_fast from Q2
group by customer_state
order by avg(diff_estimated_delivery - time_to_delivery) desc
limit 5),

Q4 as
(select customer_state, round(avg(diff_estimated_delivery - time_to_delivery),2) as delivery_speed,
dense_rank() over(order by (avg(diff_estimated_delivery - time_to_delivery)) ) as rank_value,
"slow" slow_or_fast from Q2
group by customer_state
order by avg(diff_estimated_delivery - time_to_delivery)
limit 5)

select * from Q3
union all
select * from Q4
```



Row	customer_state	delivery_speed	rank_value	slow_or_fast
1	AC	20.98	1	fast
2	RO	20.04	2	fast
3	AM	19.93	3	fast
4	AP	18.4	4	fast
5	RR	18.33	5	fast
6	AL	8.74	1	slow
7	MA	9.91	2	slow
8	SE	10.0	3	slow
9	ES	10.65	4	slow
10	BA	10.98	5	slow

Sellers in Country like AL, MA, SE, ES, BA should improve their delivery speed and try to meet the given estimated delivery time.

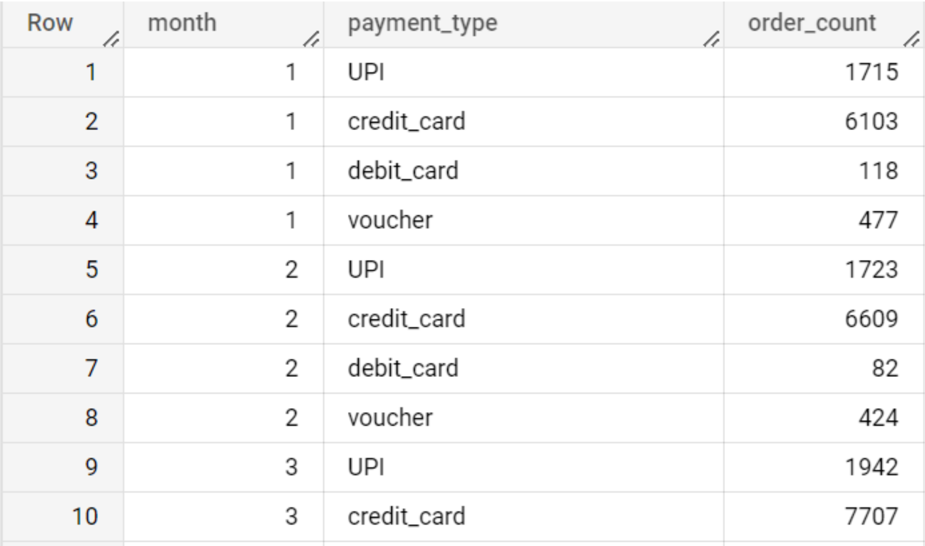
Q6. Payment type analysis:

a. Month over Month count of orders for different payment types

Ans:

```
with Q1 as
(select extract (month from o.order_purchase_timestamp) as month, o.order_id, p.payment_type
from `Target_store.orders` o join `Target_store.payments` p on o.order_id = p.order_id)

select month, payment_type, count(order_id) order_count from Q1
group by month, payment_type
order by month, payment_type
```



Row	month	payment_type	order_count
1	1	UPI	1715
2	1	credit_card	6103
3	1	debit_card	118
4	1	voucher	477
5	2	UPI	1723
6	2	credit_card	6609
7	2	debit_card	82
8	2	voucher	424
9	3	UPI	1942
10	3	credit_card	7707

By observing data, most of the orders were done by credit card, as they get benefits of EMIs and credit points on spending. Seconds highest orders were taken by UPI, as it is a fastest ways of payment method, Voucher are type of discounts which attracts people.

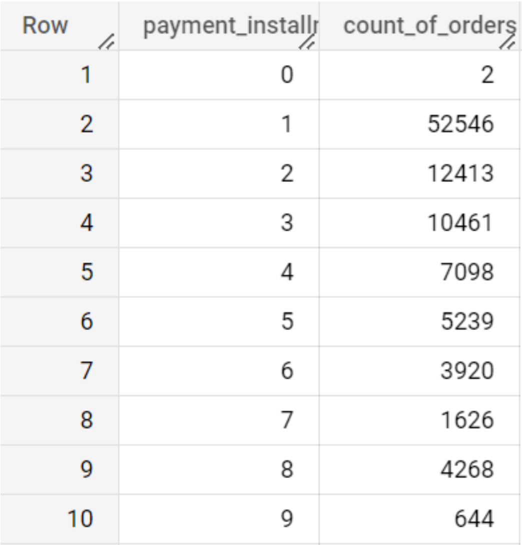
Suggetion: Increase Vouchers and give more offers on UPI payment, there is scope of increase in transaction over UPI

****

b. Count of orders based on the no. of payment instalments

Ans:

```
select  p.payment_installments , count(o.order_id) as count_of_orders
from `Target_store.orders` o join `Target_store.payments` p on o.order_id = p.order_id
group by p.payment_installments
order by p.payment_installments
```



Row	payment_install	count_of_orders
1	0	2
2	1	52546
3	2	12413
4	3	10461
5	4	7098
6	5	5239
7	6	3920
8	7	1626
9	8	4268
10	9	644

There are majority of peoples ordering using EMI option, may be due to ease of credit card use.